IV.3 (a) Definition of the Turing Machine

For this subsection no additional material has been added yet.
IV.3 (b) Equivalence of URM computable and Turing computable functions

Formal Lemma URM-computable $\Rightarrow$ TM-computable

Lemma (3.4)

If $f : \mathbb{N} \rightarrow \mathbb{N}$ is URM-computable then it is Turing-computable by a TM with alphabet $\{0, 1, \sqcup, \sqcap\}$.

Remark

The proof that every Turing computable function is URM computable will not be given in this Section.

(It could be done directly. A much nicer argument which makes use of the notion of partial recursive functions can be found in the notes of “Computability Theory”.)

Notation: $\tilde{\text{bin}}$

In this proof we will represent a configuration of a URM by a sequence of possibly non-normalised strings on the tape representing the registers.

So we want to get a short notation for “The tape contains $s_0 \sqcup n_1 \squint s_2 \sqcup \cdots \sqcup s_k$ where $s_i$ is a binary representation of $n_i$” (where $n_i$ is the simulated content of register $R_i$).

We define $\tilde{\text{bin}}(n)$ as one of the binary representations of $s$.

Then we can write for the above:

“The tape contains $\tilde{\text{bin}}(n_0) \sqcup \tilde{\text{bin}}(n_1) \sqcup \cdots \sqcup \tilde{\text{bin}}(n_k)$”.

So $\tilde{\text{bin}}(n)$ denotes one of the possible choices for strings $s$ s.t. $(s)_2 = n$.

- So $\tilde{\text{bin}}(1)$ can be "1", "01", "001", etc.
- In the special case 0 we treat the empty string as one of the possible representations, so $\tilde{\text{bin}}(0)$ can be "", "0", "00", "000", etc.

Proof of Lemma 3.4

- When carrying out intermediate calculations, it is easier to refer to $\tilde{\text{bin}}(n)$ rather than $\text{bin}(n)$
  - E.g. we can set a number on the tape easily to an element of $\tilde{\text{bin}}(0)$ by overwriting it with 0s.
  - In order to set it to $\text{bin}(0)$ one would need to make sure that exactly one 0 remains. Then one usually has to shift left the content of the tape to the right of the original number.

Notation

The tape of a TM contains $a_0, \ldots, a_I$ means:

- Starting from the head position, the cells of the tape contain $a_0, \ldots, a_I$.
- All other cells contain $\sqcup, \sqcap$. 

CS 236

Sect. IV.3 (b)
Proof of Lemma 3.4

Assume
- \( f = U(n) \),
- \( U \) refers only to \( R_0, \ldots, R_{l-1} \) and \( l > n \).

We define a TM \( T \), which simulates \( U \). Done as follows:
- That the registers \( R_0, \ldots, R_{l-1} \) contain \( a_0, \ldots, a_{l-1} \) is simulated by the tape containing \( \tilde{\text{bin}}(a_0) \_ \ldots \_ \tilde{\text{bin}}(a_{l-1}) \).
- An instruction \( I_j \) will be simulated by states \( s_{j,0}, \ldots, s_{j,i} \) with instructions for those states.

Example

Assume the URM is about to execute instruction
- \( I_4 = \text{pred}(2) \) (i.e. \( \text{PC} = 4 \)),
- with register contents
  \[
  \begin{array}{c|c|c}
  R_0 & R_1 & R_2 \\
  \hline
  2 & 1 & 3 \\
  \end{array}
  \]
- Then the URM will end with
  - \( \text{PC} = 5 \)
  - and register contents
    \[
    \begin{array}{c|c|c}
    R_0 & R_1 & R_2 \\
    \hline
    2 & 1 & 2 \\
    \end{array}
    \]

Example

Then we want that, if the simulating TM is
- in state \( s_{4,0} \),
- with tape content \( \tilde{\text{bin}}(2) \_ \tilde{\text{bin}}(1) \_ \tilde{\text{bin}}(3) \)
- it should reach
  - state \( s_{5,0} \)
  - with tape content \( \tilde{\text{bin}}(2) \_ \tilde{\text{bin}}(1) \_ \tilde{\text{bin}}(2) \)
Proof of Lemma 3.4

Furthermore, we need initial states $s_{\text{init},0}, \ldots, s_{\text{init},j}$ and corresponding instructions, s.t.

- if the TM initially contains
  
  \[ \overline{\text{bin}}(b_0) \overline{\text{bin}}(b_1) \cdots \overline{\text{bin}}(b_{n-1}) \]

- it will reach state $s_{0,0}$ with the tape containing

\[ \overline{\text{bin}}(b_0) \overline{\text{bin}}(b_1) \cdots \overline{\text{bin}}(b_{n-1}) \overline{\text{bin}}(0) \cdots \overline{\text{bin}}(0) \]

$\ell - n$ times

Example

Consider the URM program $U$ (which was discussed already in the section on URMs):

$\begin{align*}
  I_0 &= \text{ifzero}(0, 3) \\
  I_1 &= \text{pred}(0) \\
  I_2 &= \text{ifzero}(1, 0) \\
  U^{(1)}(a) &\simeq 0.
\end{align*}$
Example

\[ I_0 = \text{ifzero}(0, 3) \]
\[ I_1 = \text{pred}(0) \]
\[ I_2 = \text{ifzero}(1, 0) \]

We saw in the last section that a run of \( U^{(1)}(2) \) is as follows:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>( R_0 )</th>
<th>( R_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_0 )</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_1 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( I_2 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( I_0 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( I_3 )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

URM Stops

---

Proof of Lemma 3.4

If we have defined this we have

- If

\[
U^{(n)}(a_0, \ldots, a_{n-1}) \downarrow ,
U^{(n)}(a_0, \ldots, a_{n-1}) \simeq c ,
\]

then \( U \) eventually stops with \( R_i \) containing some values \( b_i \), where \( b_0 = c \).

Then, the TM \( T \) starting with \( \text{bin}(a_0) \ldots \text{bin}(a_{n-1}) \)
will eventually terminate in a configuration

\[
\text{bin}(b_0) \ldots \text{bin}(b_{k-1}) .
\]

for some \( k \geq n \).

Therefore \( T^{(n)}(a_0, \ldots, a_{n-1}) \simeq b_0 = c \).
Proof of Lemma 3.4

- It follows $U(n) = T(n)$,
  and the proof is complete, if the simulation has been introduced.
- The following slides contain a detailed proof, which will not be presented in the lecture this year.

Informal description of the simulation of URM instructions.

- **Initialisation.**
  Initially, the tape contains $\text{bin}(a_0) \cdots \text{bin}(a_{n-1})$.
  We need to obtain configuration:
  $$\text{bin}(a_0) \cdots \text{bin}(a_{n-1}) \cdot \text{bin}(0) \cdots \cdot \text{bin}(0).$$
  Achieved by
  - moving head to the end of the initial configuration
  - inserting, starting from the next blank, $l - n$-times $0$,
  - then moving back to the beginning.

- **Simulation of URM instructions.**
  - **Simulation of instruction** $I_k = \text{succ}(j)$.
    Need to increase $(j + 1)$st binary number by 1
    Initial configuration:
    $$\text{bin}(c_0) \cdots \text{bin}(c_j) \cdots \text{bin}(c_l)$$
    "\text{↑} $s_{k,0}$"  
    - First move to the $(j + 1)$st blank to the right. Then we are at the end of the $(j + 1)$st binary number.
    $$\text{bin}(c_0) \cdots \text{bin}(c_j) \cdots \text{bin}(c_l)$$
    "\text{↑}"
    - Now perform the operation for increasing by 1 as above.
      At the end we obtain:
      $$\text{bin}(c_0) \cdots \text{bin}(c_j + 1) \cdots \text{bin}(c_l)$$
      "\text{↑}"
    - It might be that we needed to write over the separating blank a 1, in which case we have:
      $$\text{bin}(c_0) \cdots \text{bin}(c_{j-1}) \text{bin}(c_j + 1) \cdots \text{bin}(c_l)$$
      "\text{↑}"
Proof of Lemma 3.4

In the latter case, shift all symbols to the left once left, in order to obtain a separating \( \downarrow \) between the \( i \)th and \( i - 1 \)st entry.

We obtain
\[
\tilde{\text{bin}}(c_0) \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \tilde{\text{bin}}(c_{j-1}) \downarrow \tilde{\text{bin}}(c_j + 1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow
\]

Otherwise, move the head to the left, until we reach the \( (j + 1) \)st blank to the left, and then move it once to the right.

We obtain
\[
\tilde{\text{bin}}(c_0) \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_{j-1}) \downarrow \tilde{\text{bin}}(c_j + 1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow
\]

\[\text{CS}\] 236
Sect. IV.3 (b) 26/ 1

IV.3 (b) Equivalence of URM computable and Turing computable functions

\[\text{CS}\] 236
Sect. IV.3 (b) 26/ 1

\[\text{CS}\] 236
Sect. IV.3 (b) 26/ 1

Proof of Lemma 3.4

Simulation of instruction \( I_k = \text{pred}(j) \).

Assume the configuration at the beginning is:
\[
\tilde{\text{bin}}(c_0) \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \tilde{\text{bin}}(c_j) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow
\]

We want to achieve
\[
\tilde{\text{bin}}(c_0) \downarrow \tilde{\text{bin}}(c_1) \downarrow \cdots \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow
\]

Done as follows:

Initially: \( \tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow \)

Finally: \( \tilde{\text{bin}}(c_0) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_j - 1) \downarrow \cdots \downarrow \tilde{\text{bin}}(c_l) \uparrow \)

Move to end of the \( (j + 1) \)st number.

Check, if the number consists only of zeros or not.

If it consists only of zeros, \( \text{pred}(j) \) doesn't change anything.

Otherwise, number is of the form \( b_0 \cdots b_{l'} 00 \cdots 0 \).

Replace it by \( b_0 \cdots b_{l'} 11 \cdots 1 \).

\[l' \text{ times}\]

Done as for succ.
IV.3 (b) Equivalence of URM computable and Turing computable functions

Proof of Lemma 3.4

- **Simulation of instruction** $I_k = \text{ifzero}(j, k').$
  - Move to $j + 1$st binary number on the tape.
  - Check whether it contains only zeros.
    - If yes, switch to state $s_{k',0}$.
    - Otherwise switch to state $s_{k+1,0}$.

This completes the simulation of the URM U.

Halting Problem with no Inputs

Theorem (3.8)

*It is undecidable, whether a Turing machine started with a blank tape terminates.*

- **Proof:**
  - Let $\text{Halt}'(e) :\Leftrightarrow e$ is a code for a Turing machine T and T started with a blank tape terminates
  - Assume $\text{Halt}'$ were decidable.
  - Then we can decide $\text{Halt}(e, n)$ as follows:
    - Assume inputs $e$, $n$.
    - If $e$ is not a code for a Turing machine, we return 0.
    - Otherwise, let $\text{encode}(T) = e$.
    - Define a Turing machine $V$ as follows:
      - $V$ first writes $\text{bin}(n)$ on the tape and moves head to the left most bit of $\text{bin}(n)$.
      - Then it executes the Turing machine T.
  - We have
    - $V$, run with blank tape, terminates
      iff $T$ run with tape containing $\text{bin}(n)$ terminates
      iff $T^{(1)}(n)\downarrow$
      iff $\{e\}(n)\downarrow$. 
**IV.3 (c) Undecidability of the Turing Halting Problem**

**Halting Problem with no Inputs**

V, run with blank tape, terminates iff \( \{e\}(n) \downarrow \).

- Let \( \text{encode}(V) = e' \). Then
  
  \[ \text{Halt}'(e') \Leftrightarrow \text{Halt}(e, n) \]

- Therefore using the decidability of \( \text{Halt}' \) we can decide \( \text{Halt}(e, n) \).
- So we have decided \( \text{Halt} \), a contradiction.

---

**IV.4 The Church-Turing Thesis**

**No Additional Material**

For this subsection no additional material has been added yet.